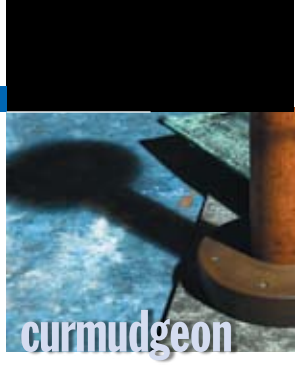

The Code Delusion

Stan Kelly-Bootle, Author

The real,

THE ABSTRACT, AND

THE PERCEIVED

No, I'm not cashing in on that titular domino effect that exploits best sellers. The temptations are great, given the rich rewards from a gullible readership, but offset, in the minds of decent writers, by the shame of literary hitchhiking. Thus, guides to the Louvre become *The Da Vinci Code Walkthrough for Dummies*, milching, as it were, several hot cows on one cover. Similarly, conventional books of recipes are boosted with titles such as *The Da Vinci Cookbook—Opus Dei Eating for the Faithful*. Dan Brown's pseudofiction sales stats continue to amaze, cleverly stimulated by accusations of plagiarism and subsequent litigation (Dan found not guilty). One strange side effect of his book's popularity and ubiquity is to hear the title shortened in conversations such as this overheard at a Microsoft cafeteria:

"Surely you've read *The Code*?"

"Not yet, but I really must—the reviews have been quite mixed."

Not yet matching the Dan-Brown Effect is the growing reaction triggered by Richard Dawkins's unholy, high-rolling *The God Delusion,*[1] including the McGraths' refutation, *The Dawkins Delusion*?, with its clever hint of theist irritation.[2] Again, the "Delusion" in my title here may suggest some desire to exploit the Dawkins celebrity, but the last thing I want is to have the Curmudgeon column dragged into overt theological speculation and controversy. I know from experience that even quasi-theological remarks on the relative merits of rival programming languages and methodologies can lead to fisticuffs and that most dreaded of responses, "Cancel my subscription."

Rather, my emerging theme and title were prompted by several "Java Puzzlers" from Joshua Bloch, Neal Gafter, et al., which touch on concepts highly relevant to the secular side of programming—namely, perception and "code illusions." In fact, I was tempted to call this column "Want to Buy Some Illusions?", which for full impact must be intoned by a husky Marlene Dietrich flashing her fishnets.[3]

It's useful to compare transitory, curable code *illusions* with the deeper, damned-near intractable code *delusions*. As Ambrose Bierce said, "Delusion is the father of a most respectable family, comprising Enthusiasm, Affection, Self-denial, Faith, Hope, Charity and many other goodly sons and daughters" (*The Devil's Dictionary*). Self-denial seems to be the prime offspring. I suspect that we dismiss as delusions all those contrary opinions held by others, but resent that accusation in the opposite direction. Illusions, by contrast, seem more amenable to rational discourse. We are amused by Houdini's tricks and long to learn how we've been fooled. Similarly, when we persist in reading a line of Java as though it means what it *ought* to mean, we are positively relieved and grateful when Joshua Bloch shatters our "code illusion." We become better programmers as a result, striving to avoid the same illusions in our own programs.

The beliefs that "software is engineering" and "software is *not* engineering" are beyond illusions. They have been labeled dangerous delusions, both being blamed for the big mess called IT. "How bad is IT?" ask those reviewing the world's largest computer project—a $24 billion budget being frittered away to unify the UK's National Health Service medical records and appointment systems. They reckon that 40,000 patients die each year through diverse, avoidable medical and administrative mistakes. Many of these can undoubtedly be avoided by improved, integrated computer systems, so the government and public concern over current delays is understandable. Successive waves of consultants, each fully versed in the latest methodologies and abstractions (they have framed certificates to prove it), urge radical revisions (the most common being severe reductions to an impossibly grandiose expectation) or even a fresh restart from scratch.[4]

Speaking of the superstitious magic of silver bullets, I was also tempted to exploit the super-best-sellers-of-all-time, known as the Harry Potter-boilers. Alas, one of my grandsons, computer-graphicist Luke Bailey, is employed by the Harry Potter filmmakers. This is the customary "declaration of possible interest conflicts," which is a custom much neglected in our fair trade. I must write to ACM's legal-eagle columnist, Pamela Samuelson.[5]

Continued on page 49

Continued from page 52

The ethics of software development is a perennial topic. Reaching me today is Peter A. Freeman's comment in his Inside Risks column ("Risks are Your Responsibility," *Communications of the ACM*, June 2007) declaring that professionals at all levels of the software development hierarchy have a *moral* [his emphasis] duty to "ensure that their results are as risk-free as possible."

That recurring "as X as possible," where X stands for any of our many desirable predicates in life, has become my latest, grumpiest pet peeve, overtaking both "appropriate" and "multiparadigmaticity" by a nose. "As X as possible" feels so feel-good that we would be churlish not to genuflect. Questioning the implied do-your-best advice marks you as someone willing to settle for less than an assumed optimal outcome. The big questions go a-begging, bowl in hand, seldom winning a serious handout.

The ineluctable challenge remains: what are the realistically achievable ranges of the variable X? Indeed, what are possible (never mind acceptable) values for the parameter X = risk-free in the hurly-burly of our specialist-labor-divided, component-based efforts? We can dream of zero-defects and zero-tolerance, but there's no computable function here converging to predictable minima. Nor any way of quantifying how a proposed smart fix here or there may smooth the path to Nirvana. Our industry, certainly the marketeering front line, shuns the slogan, "Least imperfect!" Claims of being "more robust than version 3.141" certainly hint that earlier bugs have been nailed, yet leaves unsaid what everyone knows. As Confucius puts it: "Program without Bug like Dog without Flea." Then he rubs in that aphorism with another: "Dog without Flea like Man without Lie." Cretan logicians will ponder whether Confucius the Man can be trusted.

My hero and fellow veteran, David Lorge Parnas (Forum, *Communications of the ACM*, June 2007), responding to Jeff Kramer's article, "Is Abstraction the Key to Computing?" (*Communications of the ACM*, April 2007), invokes Einstein's aphorism: "Everything should be as simple as possible but not simpler." In the context of using abstractions to build models of reality, Parnas concludes, "Finding the simplest model that is not a *lie* [my emphasis] is the key to better software design." Kramer and Parnas agree on the central importance of Abstraction (worthy of Capitalization!) but differ on definitions. Kramer's Abstraction stresses the need to temporarily hide distracting[6] details, else you are too bogged down to make a start. Parnas prefers Dijkstra's definition: "An abstraction is one thing that represents several real things equally well."

Usefully merging the two related ideas, we still hit some epistemological brick walls. First, how to assess the "wellness" of particular abstract representations—especially when the expert abstracters disagree, often violently, as to what can be safely hidden? Recall the wonderful treatise on improving milk production: "Let C be a perfectly uniform, spherical cow." Compare, too, the historical Algol 68 squabbles where Hoare and Dijkstra accused van Wijngaarden and his ilk of mis-abstractions and complexifying formalisms. Surely computer-language design should be the key test bed for guru theories on the effective and provable deployment of Abstractions and Models. As Parnas points out, Dijkstra's views on Abstraction have been around for 40 years and "underlie every software development method proposed since then." Is this rather like saying, "Christianity is a marvelous thing if only someone tried it?"

Second, accusing an inadequate Abstraction of being a lie is rather circular and hard to parse unemotionally without reenduring the recent decades of postmodernistic polemic on the relativity of truth. The physicist's Standard Model is hardly an inadequate lie. For known experimental-observational reasons, the "real things" out there are deduced from fleeting ghost tracks. You might almost say that "models create (reify) our reality" rather than the reverse. I'm a Platonist on odd-numbered days, so at the moment I do believe in real worlds of both thingies and ideas out there begging to be perceived and modeled.

Third, outside the lecture hall, frustrating our desire to build "truthful" software models, we have that old devil of requirements: volatility. Our products sink or swim in chaotic seas of arbitrary change beyond the reach of what we casually call "flexible and extensible systems." Changes creep in at all levels, with different impacts as we move up or down the chain. Freeman highlights the responsibility challenge when creating components. Of course, components must fulfill their agreed-upon contracts, often relying on subcomponents keeping *their* promises. It's turtles all the way down, to borrow Bertrand Russell's cosmogenic anecdote. Would capital punishment help? Mockery and job dismissals have failed. For those in charge of the big picture, seeing that the components are glued (hardly a reassuring idiom) together properly, Freeman moots some high-level software analysis tools that may help. These, in turn, will need the most reliable components and assemblage available. Metaturtles all the way down? Furthermore, our code runs (eventually!) on hardware that has been known to change in annoyingly discontinuous steps without due process.

It used to be said in the 1960s (by me, no less) that

the UK tax system was deliberately devised to crush the hubris of the programmer. The IRS (dubbed the Internal *Revision* Service) was an evil cabal of tax collectors (descendants of Levi before he saw the light as Matthew the disciple [Luke 5:27-32]) who diabolically timed their tax-code changes to produce maximum disruption to the payroll software. The latter was scrutinized by the cabal and when the software appeared to have mastered the previous tax-code quirks, cunning revisions would be introduced. These were never simple rate or level changes that could be easily accommodated by resetting global constants. Changes were crafted to strike at the very axioms and structure of existing programs.

Inside the insulated CS lecture hall (I pause to insert the weaselly "there are exceptions"), we do find awareness of and useful debate on that other chief impediment to achieving "as X as possible" for given values of X. This pertains to the irreducible imprecision of NL (natural language), upon which all our requirements specifications ultimately depend. I say this with all due respects to my formalist friends. (See this column, "The Calculus Formally Known as Pi," May 2006). It impacts even the most fixed, formal specifications untroubled by external events such as the whims of the tax-persons. Van Wijngaarden context-free grammars pretend to eschew "ambiguity-prone, context-ridden NL prose," yet the Algol 68 committees and compiler-writers were/are forced to conduct their debates in NL (German or Dutch being preferred for dominating precision). Both Dijkstra and Parnas stressed the importance and dangers of NL. With effort, the grosser ambiguities, at least in our native tongue[s], can be avoided. Those who write and read specifications must remain on guard and confer "as much as possible!" Those who program, as Parnas preached, must not be misled by a variable named NetPay. The program is UL (*un*natural language). The semantics of NetPay are defined by how the underlying bit patterns are manipulated (see this column, "Linguae Francae," December/January 2004/2005).

The mysterious abstraction that powers all our communication is natural language itself. Whether it meets Dijkstra's criterion of "representing several real things equally well" can only, frustratingly, be debated using NL (and perhaps some waving of the hands).

READER FEEDBACK

Responses to my plea for quotable quirks are arriving. From long-faithful, oft-suffering reader Michael D. Zorn:

"I'm delighted to hear you're now the Curmudgeon di tutti Curmudgii.

"For howlers and assorted crass annoyances, I've been getting the usual unsolicited inspirational e-mail from a company or two that offer "webinars" on sundry topics. Maybe it's just me, but that word grates on the sensibilities like fingernails on a genuine slate blackboard. Needless to say, I ignore their offers. A Google search of *webinar* turns up 8.9 million hits. Even allowing for the usual duplication, that's far too many. One of the first is a helpful definition from a Web site called Webopedia. More and more I think maybe the French have the right idea: build a wall of impenetrable armor around their language.

"A discussion of 'Alloneword' (May/June 2007) could not possibly be complete without mention of those four (five?) gigantic thunderclaps in *Finnegans Wake* (I remembered this time about the apostrophe). I think I'll put them all together, apply a little gemmatria, and find out whether or not Macbeth was really the twelfth Caliph of Dunsinane."

For these and several space-precludes observations, Mike wins a prize to be named later. From George Jensen, yet another regular and esteemed e-pal:

"The stray decimal points you cite stray elsewhere. A neighborhood book club read *Cod* (meaning cash on delivery) last summer; the table of conversions, metric to English, in the back had a couple of off-by-ten errors."

As for "at one" and "atone," [preacher's name withheld] seemed to think, some years back, that "atonement" meant "unity." Well, there's glory for you. Q

REFERENCES

1. Dawkins, R. 2006. *The God Delusion*. Houghton Mifflin. Note that I use the less exalted genitive "Dawkins's" rather than "Dawkins'." The Stylish Guide gurus from Chicago allow the more euphonious genitives such as Jesus', Moses', and even Dickens'; perhaps in time Richard will justify this usage.
2. McGrath, A. and J. 2007. *The Dawkins Delusion? Atheist fundamentalism and the denial of the divine*. Society for Promoting Christian Knowledge. Note the "?" in the title, reflecting the McGraths' politeness, a trait that many find absent in Dawkins's polemic.
3. As sung in Billy Wilder's darkly satirical movie, *A Foreign Affair*, set in post-WW2 Berlin. Marlene in her *Blue Lamp* role of cabaret chanteuse offers to sell all her precious illusions "for a penny," which is, I suppose, the ultimate illusion.
4. A fascinating idiom for itchy wordsmiths. If things are not "up to scratch," one may have to "start from scratch." One might have to employ "scratch" programmers in the golfing sense of "no handicap or allowance." This tautological blend of *scat* and *cratch*

(both meaning *scratch!*) has come a long way.

5. Professor Samuelson's Legally Speaking column in *Communications of the ACM* is an ongoing source of supreme, courtly wisdom. Echoes of Portia in the Bard's *Merchant of Venice*: "A Daniel come to judgment! Yea, a Daniel. O wise young judge, how I do honour thee." (Shylock, act 4 scene 1.) Of course, Portia herself, disguised as Doctor of Laws Balthasar, is the very epitome of *not* declaring a conflict of interest—to wit, her interest in saving that wimp Bassanio. In passing, one might note that the confusion between *uninterested* and *disinterested* continues to flourish, to the point where dictionaries have given up the fight and declared the two words entirely synonymous. One learned judge was quoted as saying he was "disinterested" in spite of being the defendant's brother-in-law. Back in our "real" commercial world, the legal metaphysics of what constitutes an "interest" has engaged Professor Samuelson when, for example, she "positions herself as a spokesperson for the public interest," translating arcane white papers for plainspeakers. See: http://www.personal.umich.edu/~dditomm/cr_final_project.html.
6. It's no coincidence that *abstract*, *distract*, *extract*, and in donnish disputes *detract!* share the Latin root *trahere/tractus* (drag, pull, or draw). All carry the notion of pulling something away from something. Distractions divert your attention; abstractions remove or gloss over details, as in nonrepresentational art or as in mathematics where the integers are considered as particular members of the more abstract set of real numbers; detractions, as in "Kramer's views are far from new," tend to undermine (pull away) rivals' reputations.

LOVE IT, HATE IT? LET US KNOW

feedback@acmqueue.com or www.acmqueue.com/forums

STAN KELLY-BOOTLE (http://www.feniks.com/skb/; http://www.sarcheck.com), born in Liverpool, England, read pure mathematics at Cambridge in the 1950s before tackling the impurities of computer science on the pioneering EDSAC I. His many books include *The Devil's DP Dictionary* (McGraw-Hill, 1981), *Understanding Unix* (Sybex, 1994), and the recent e-book *Computer Language—The Stan Kelly-Bootle Reader*. *Software Development Magazine* has named him as the first recipient of the new annual Stan Kelly-Bootle Eclectech Award for his "lifetime achievements in technology and letters." Neither Nobel nor Turing achieved such prized eponymous recognition. Under his nom-de-folk, Stan Kelly, he has enjoyed a parallel career as a singer and songwriter.